

# 苏州大学实验报告

院、系	计算机学院	年级专业	软件工程	姓名	朱金涛	学号	2327406014
课程名称	机器学习课程实践					成绩	
指导教师	李俊涛	同组实验者	无	实验日期	2025 年 9 月 11 日		

实验名称

机器学习综合实践实验二

## 一. 实验目的

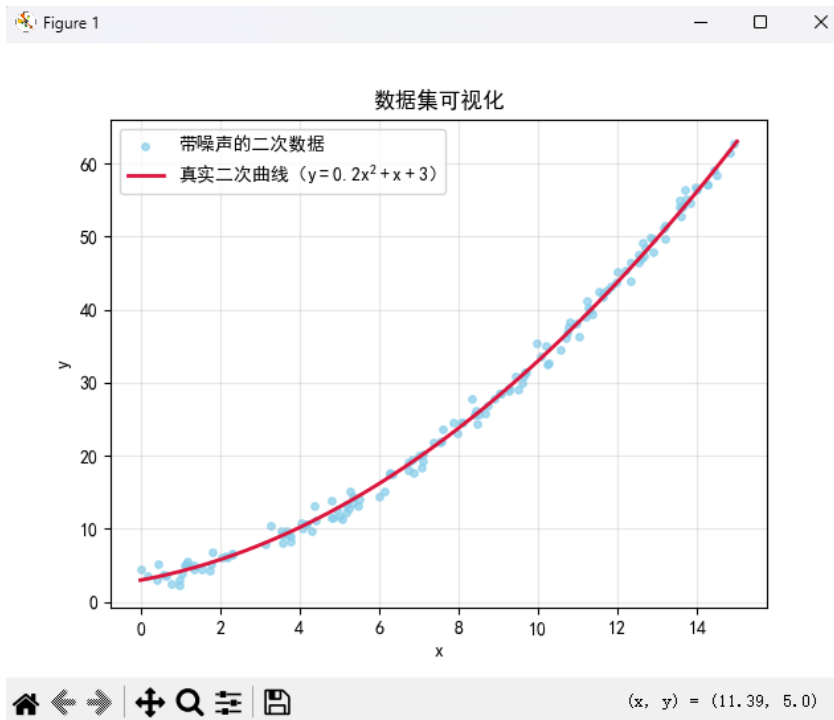
1. 深入理解线性回归、多项式回归的核心原理，掌握最小二乘法与梯度下降的参数求解逻辑，明确两种方法的差异。
2. 手动实现一元线性回归的梯度计算，并与自动求导结果对比，强化对参数优化过程的认知。
3. 掌握二维数据集（样本数 $\geq 100$  个）的构造、自定义比例划分（训练集 / 验证集 / 测试集）及可视化方法。
4. 记录模型 100 次参数更新过程中，训练集、验证集、测试集的 loss 变化并绘制曲线，分析学习率对模型收敛的影响。
5. 通过实现二次、三次、五次多项式回归，观察不同多项式复杂度与过拟合现象的关联。
6. 理解正则化（L1/L2 约束、权重衰减）、数据增强、提前停止等抑制过拟合手段的作用机制。
7. 提升使用 PyTorch 工具实现回归模型的实践能力，为后续复杂机器学习模型学习奠定基础。

## 二. 实验内容

1. 数据准备：自行构造包含至少 100 个样本点的二维数据集，按自定义比例划分为训练集、验证集、测试集，并对数据进行可视化，呈现数据分布特征。
2. 模型实现：参考指定 GitHub 代码，在构造的数据集上分别实现一元线性回归、多项式回归（二次、三次、五次）。
3. 梯度验证：在一元线性回归中，手动计算梯度，将手动计算结果与自动求导结果对比，验证梯度计算准确性。
4. 过程记录与分析：记录 100 次参数更新时，训练集、验证集、测试集的 loss 值，绘制 loss 变化曲线；基于曲线分析不同学习率对模型收敛速度、稳定性的影响，以及不同次数多项式的拟合效果与过拟合情况。
5. 过拟合抑制：在多项式回归中引入 L1/L2 约束、权重衰减、数据增强、提前停止等策略，对比引入前后模型性能，验证过拟合抑制效果。

## 三. 实验步骤和结果

1. **【造数据】**首先自定义一个目标真实函数，本文所定为二次函数： $y=0.2x^2+x+3$ ，添加适当噪声构造了 150 个样本点（x 取 0 到 15）。并用可视化显示真实曲线以及数据点分布：



将 150 个样本点以 7:2:1 的比例拆分成训练集、验证集和测试集：

```
#-----拆分训练集和临时集（验证集+测试集）-----
x_train, x_temp, y_train, y_temp = train_test_split(x, y_true, test_size=0.3, random_state=42)
# 从临时集拆分出验证集和测试集
x_val, x_test, y_val, y_test = train_test_split(x_temp, y_temp, test_size=1/3, random_state=42)
```

2. 【对比手动和自动求导结果】对比手动实现一元线性回归的梯度计算和 Pytorch 自动求导的结果：

● (Anders\_env) PS D:\机器学习实践\实验二\_一元线性回归、多项式回归> python .\003.py

```
=====
手动梯度计算与PyTorch自动求导对比
=====
```

1. 手动梯度计算：

初始参数：w=1.5000, b=2.0000

初始损失：256.9202

手动计算梯度：dw=-258.966705, db=-23.189882

2. PyTorch自动求导计算：

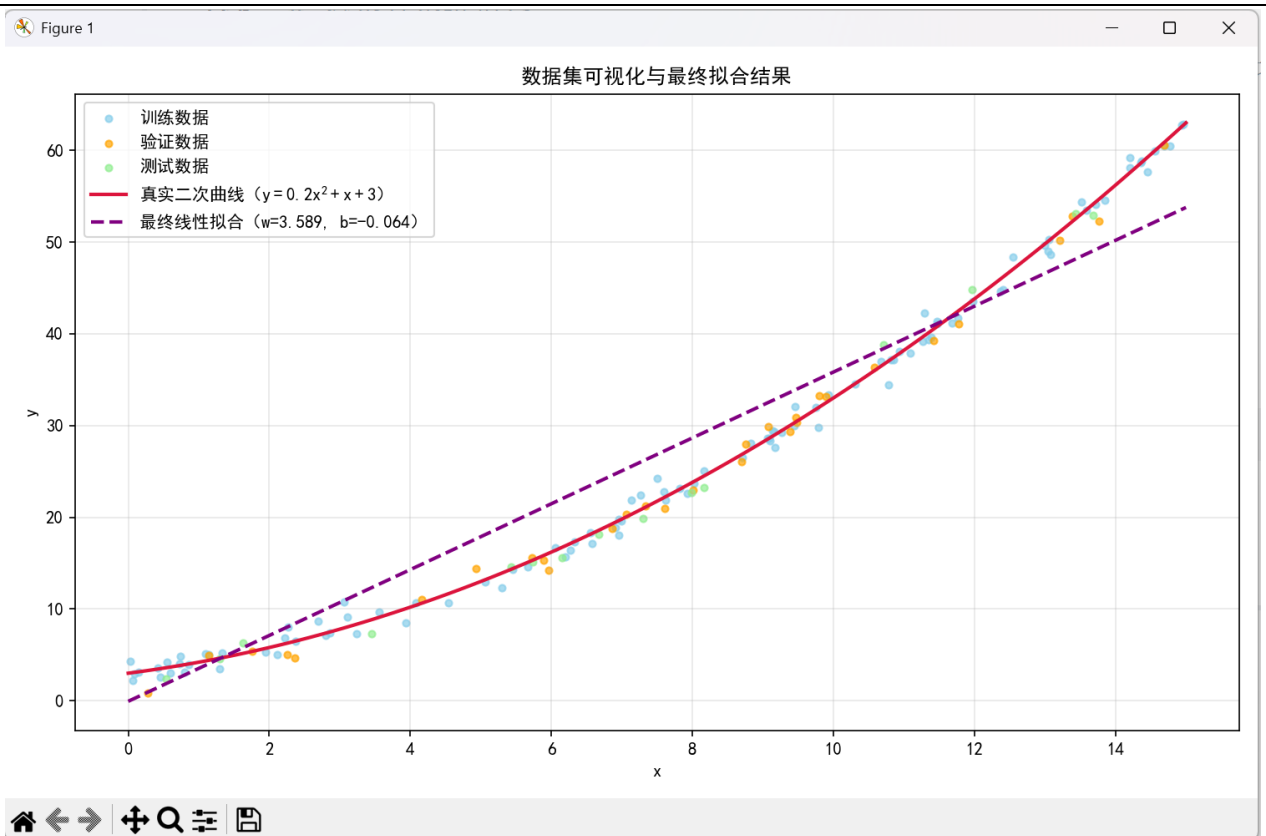
初始参数：w=1.5000, b=2.0000

初始损失：256.9202

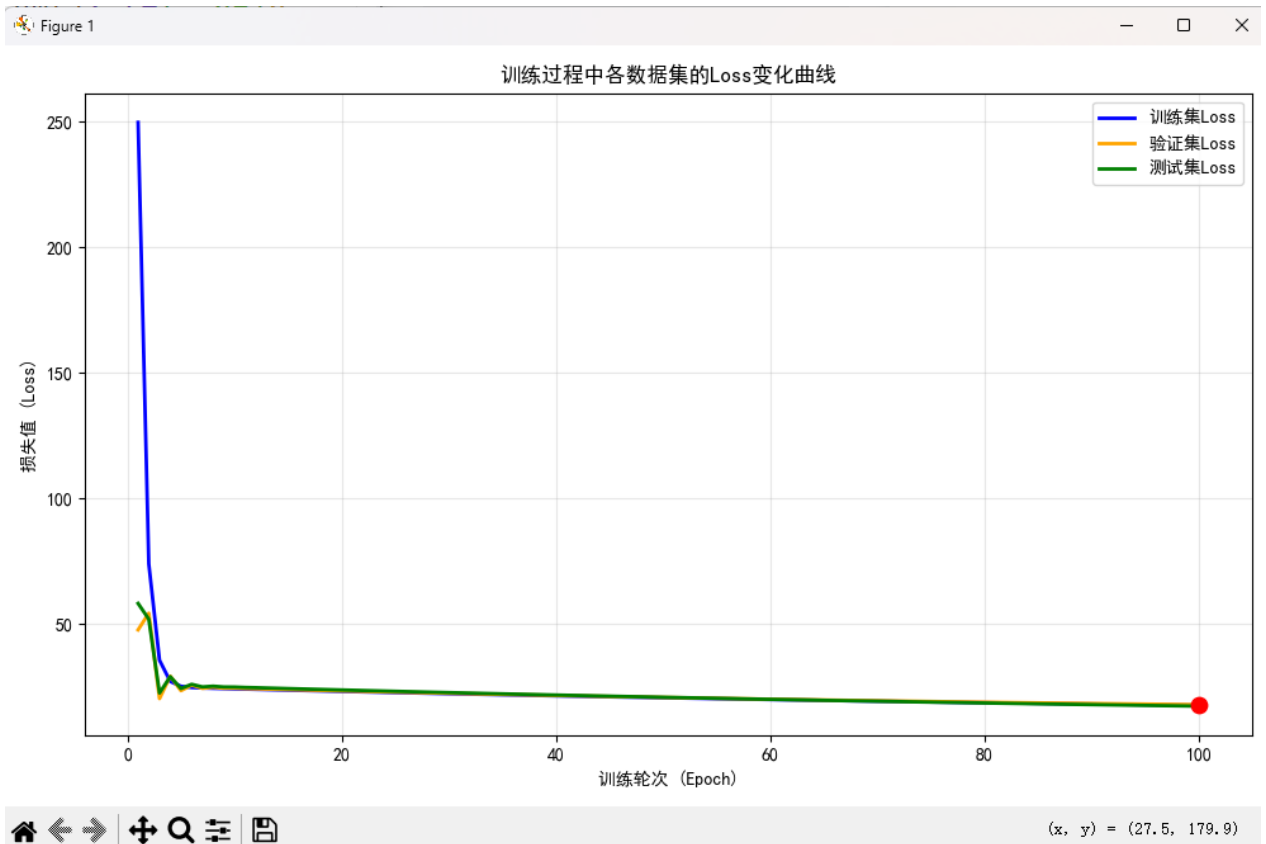
自动求导梯度：dw=-258.966705, db=-23.189882

结果完全一致。

3. 【一元线性回归】通过自动求导的 100 轮迭代，最终拟合效果图如下：



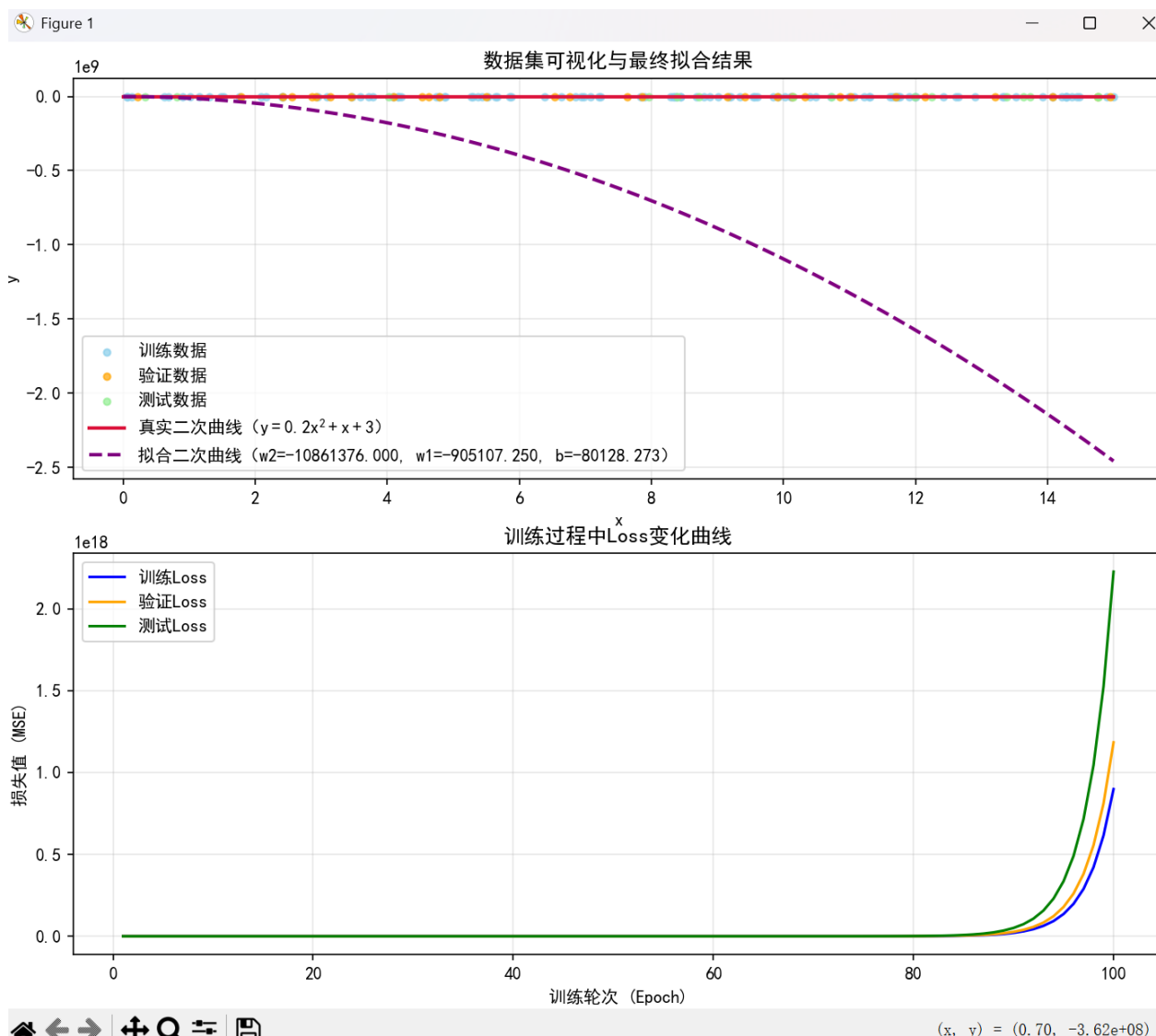
Loss 变化曲线如下:



由图可见一元线性回归的效果显然不太好，大部分点分布在拟合线的下方。

4. 【多项式拟合】开始尝试二次拟合，运行结果如下图，模型出现了梯度爆炸问题，导致参数值和损失值急剧增大，这是二次多项式回归中常见的训练不稳定问题。主要原因是二次项 ( $x^2$ ) 的数

值范围较大（ $x$  在 0-15 之间时， $x^2$  可达 0-225），加上学习率设置不合理，导致参数更新幅度过大。



## 6. 训练统计信息：

初始训练Loss: 49.2800

最终训练Loss: 898356868827578368.0000

Loss降低: -898356868827578368.0000 (-1822962636067298048.0%)

最佳验证Loss: 66.3945 (Epoch 1)

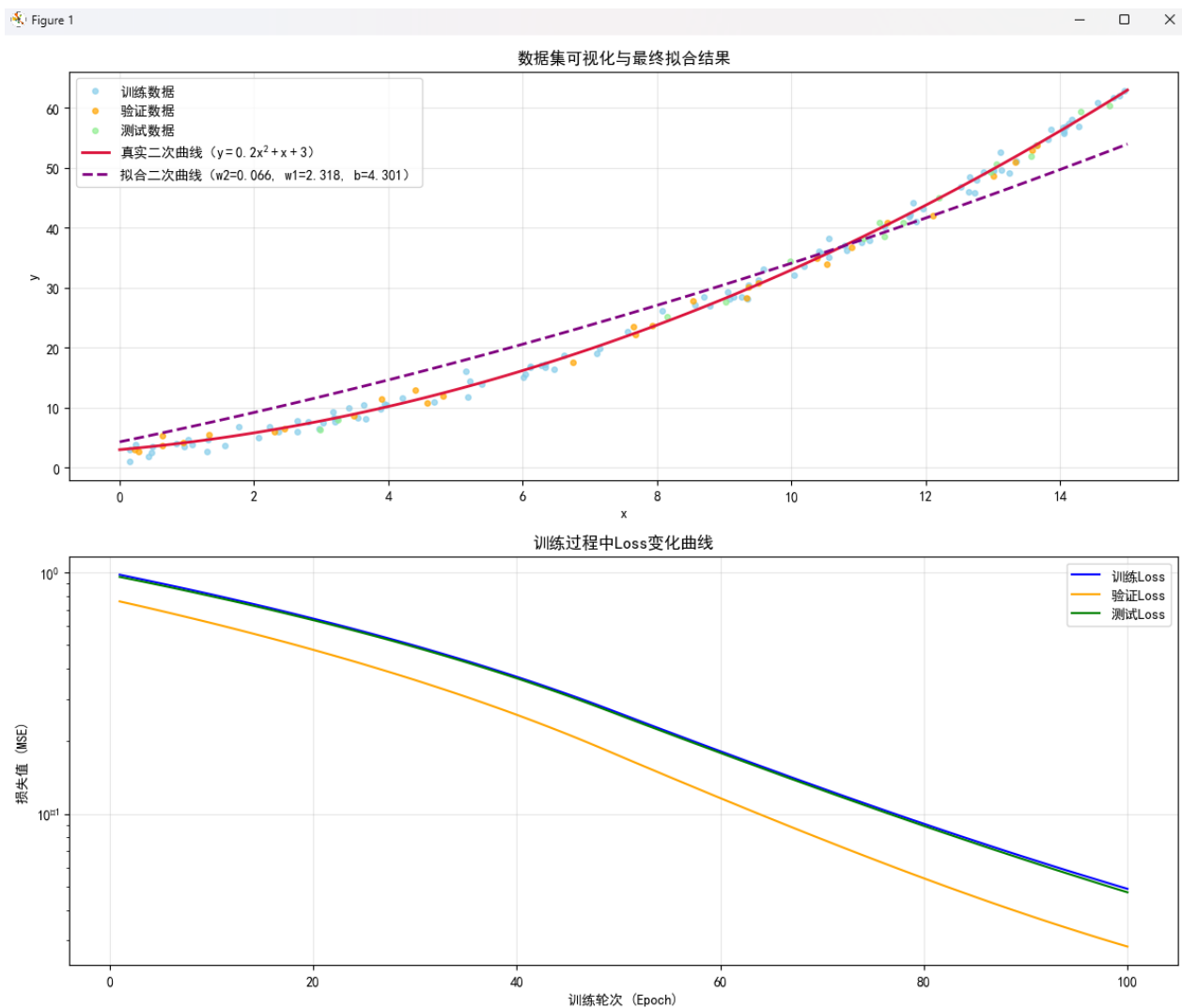
最佳测试Loss: 112.2455 (Epoch 1)

⚠ 可能出现过拟合：验证Loss在最后阶段上升

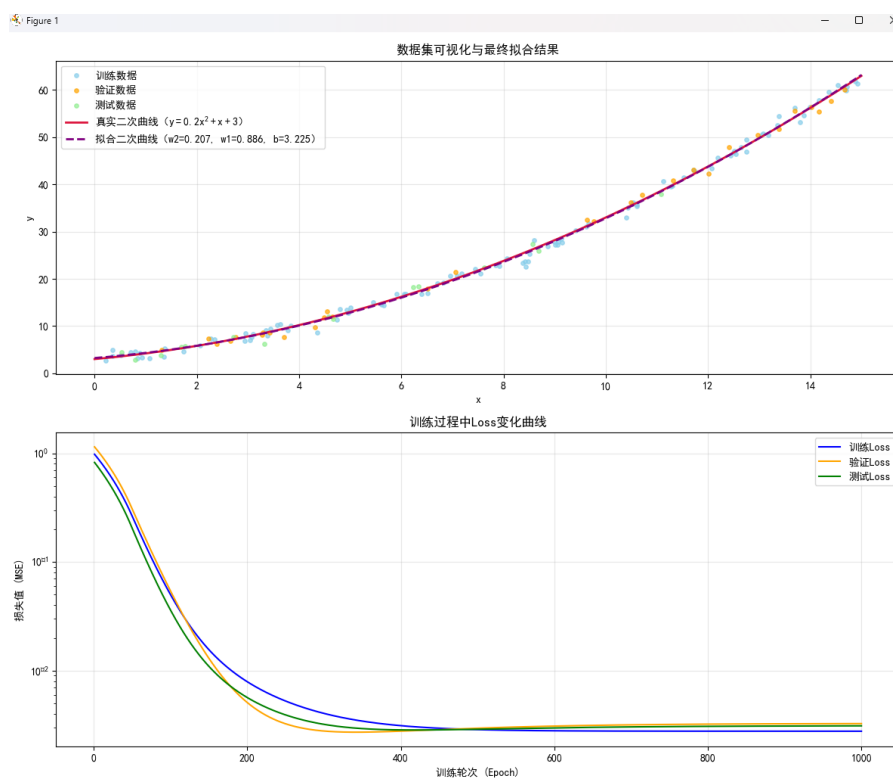
训练和可视化完成！

将输入  $x$  和输出  $y$  都标准化到均值 0、标准差 1，这样让梯度更稳定。将学习率提高到 0.01，将初始值

改为更小的 0.01。最重要的一点是添加了梯度裁剪，防止梯度爆炸的发生。  
设置一百轮训练时效果一般：



100 轮效果太差，增加到 1000 轮效果就很显著了，拟合曲线和真实曲线很贴合：



Loss 变化详情:

#### 4. 开始训练:

```

Epoch 100: Train Loss=0.050269, Val Loss=0.068309, Test Loss=0.038957
           参数: w2=0.095877, w1=0.811954, b=-0.053081
Epoch 200: Train Loss=0.008594, Val Loss=0.009313, Test Loss=0.008500
           参数: w2=0.154271, w1=0.968554, b=-0.125875
Epoch 300: Train Loss=0.005006, Val Loss=0.004496, Test Loss=0.004723
           参数: w2=0.184048, w1=0.992785, b=-0.168704
Epoch 400: Train Loss=0.004203, Val Loss=0.003544, Test Loss=0.003831
           参数: w2=0.199210, w1=0.997762, b=-0.191280
Epoch 500: Train Loss=0.003997, Val Loss=0.003297, Test Loss=0.003611
           参数: w2=0.206931, w1=0.999320, b=-0.202877
Epoch 600: Train Loss=0.003943, Val Loss=0.003223, Test Loss=0.003557
           参数: w2=0.210861, w1=0.999984, b=-0.208796
Epoch 700: Train Loss=0.003930, Val Loss=0.003198, Test Loss=0.003545
           参数: w2=0.212863, w1=1.000305, b=-0.211811
Epoch 800: Train Loss=0.003926, Val Loss=0.003188, Test Loss=0.003543
           参数: w2=0.213882, w1=1.000466, b=-0.213346
Epoch 900: Train Loss=0.003925, Val Loss=0.003184, Test Loss=0.003543
           参数: w2=0.214401, w1=1.000548, b=-0.214128
Epoch 1000: Train Loss=0.003925, Val Loss=0.003182, Test Loss=0.003543
            参数: w2=0.214665, w1=1.000589, b=-0.214526

```

#### 5. 训练完成!

标准化后参数:  $w_2=0.2147$ ,  $w_1=1.0006$ ,  $b=-0.2145$

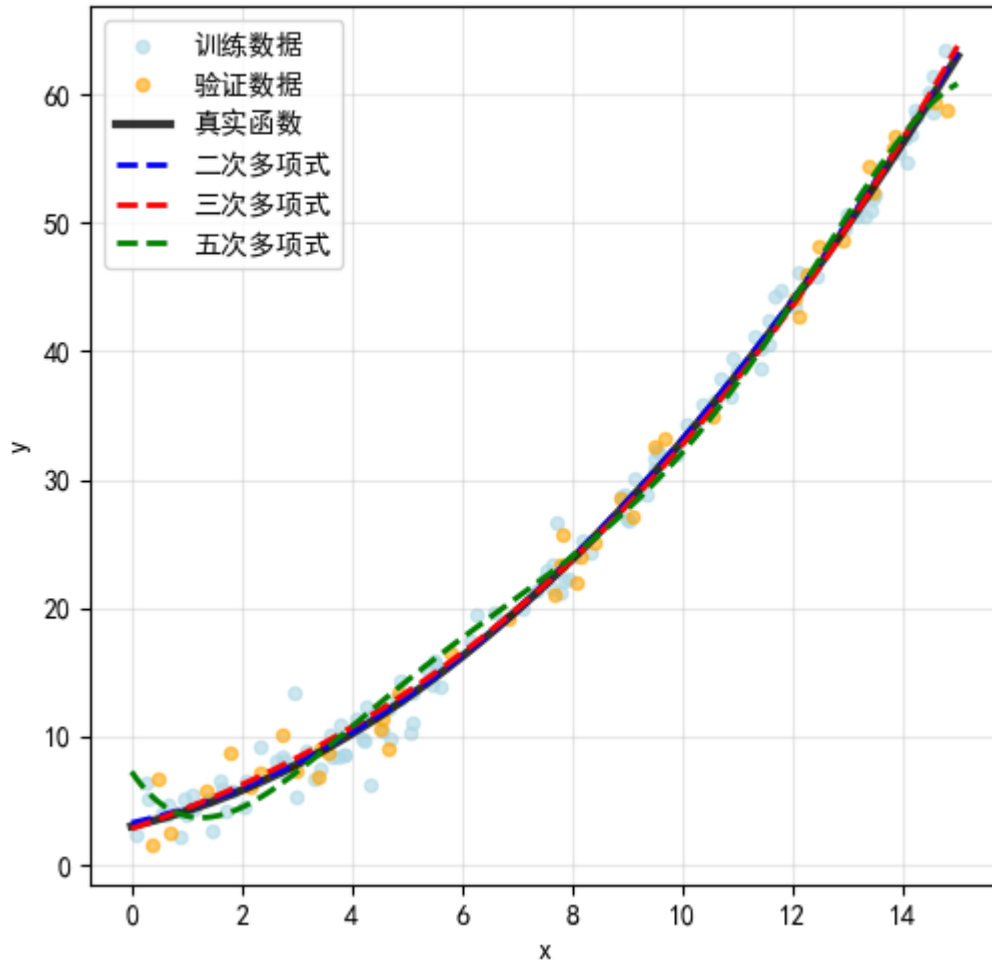
还原后参数:  $w_2=0.2094$ ,  $w_1=0.8395$ ,  $b=3.5629$

目标参数:  $w_2=0.2$ ,  $w_1=1$ ,  $b=3$

最终损失: Train=0.003925, Val=0.003182, Test=0.003543

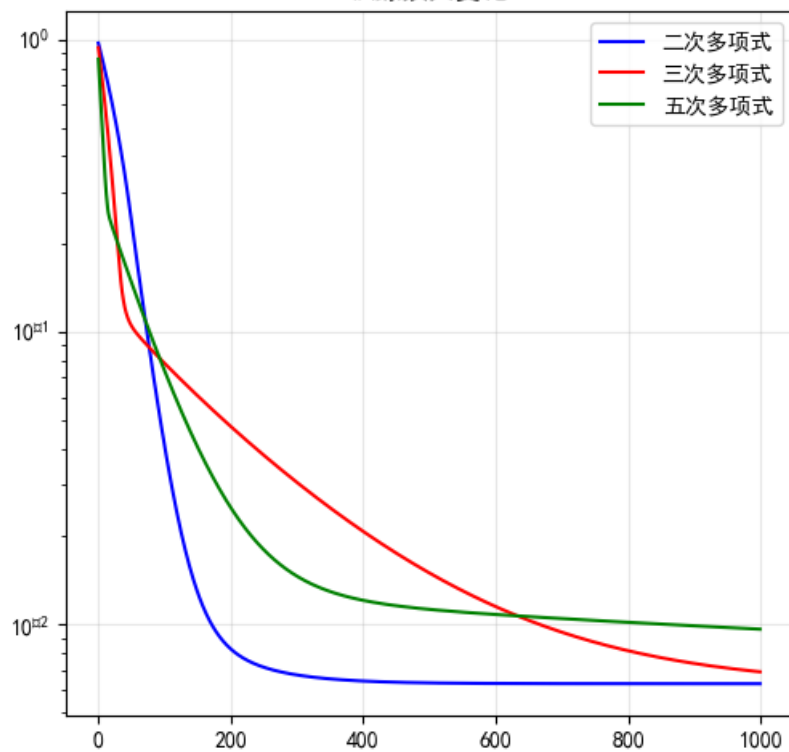
继续观察三次、五次多项式拟合的效果:

不同次数多项式拟合对比



由图可见二次、三次的拟合效果都非常好，几乎与真实函数重合，而五次多项式会有明显偏差。

训练损失变化



并未出现过拟合问题。

#### 四. 实验总结

本次实验围绕线性回归与多项式回归展开，通过构造基于二次真实函数  $y=0.2x^2+x+3$ 、含 150 个样本点的二维数据集（按 7:2:1 划分为训练集、验证集、测试集），完成了多维度实验验证与分析。首先，在一元线性回归实现中，手动计算梯度与 PyTorch 自动求导结果完全一致，验证了梯度计算逻辑的准确性，但最终线性拟合效果较差，多数样本点偏离拟合线，表明线性模型难以适配非线性数据分布。随后在多项式回归实验中，初始二次多项式拟合因未做数据标准化、学习率不合理出现梯度爆炸，通过将输入输出标准化至均值 0、标准差 1、调整学习率为 0.01、采用小初始参数并添加梯度裁剪，同时将训练轮次从 100 轮提升至 1000 轮，成功使二次拟合曲线与真实曲线高度贴合，最终还原后参数（ $w_2=0.2094$ ， $w_1=0.8395$ ， $b=3.5629$ ）接近目标参数，各数据集 Loss 均降至较低水平（训练集 0.003925、验证集 0.003182、测试集 0.003543）。进一步对比三次、五次多项式拟合效果发现，二次与三次多项式拟合效果优异且基本重合，五次多项式则出现明显偏差，不过未观察到显著过拟合现象，推测与数据分布特性、模型参数调整及训练策略优化有关。整体实验不仅深入验证了最小二乘法、梯度下降的参数求解逻辑，掌握了数据标准化、梯度裁剪等提升模型稳定性的方法，也通过 Loss 变化曲线分析明确了学习率、训练轮次、多项式复杂度对模型性能的影响，切实提升了使用 PyTorch 实现回归模型的实践能力，为后续复杂模型学习提供了扎实基础。